STUDS AND IMMIGRANTS IN MULTIRECOMBINED EVOLUTIONARY ALGORITHM TO FACE WEIGHTED TARDINESS SCHEDULING PROBLEMS

Pandolfi D., De San Pedro M., Villagra A, Vilanova G.
Proyecto UNPA-29/B032[1]
División Tecnología
Unidad Académica Caleta Olivia
Universidad Nacional de La Patagonia Austral
Ruta 3 Acceso Norte s/n
(9011) Caleta Olivia – Santa Cruz - Argentina
e-mail: {edesanpedro,dpandolfi,avillagra}@uaco.unpa.edu.ar; gvilanov@satlink.com
Phone/Fax : +54 0297 4854888

Gallard R.
Laboratorio de Investigación y Desarrollo en Inteligencia Computacional (LIDIC)[2]
Departamento de Informática
Universidad Nacional de San Luis
Ejército de los Andes 950 - Local 106
(5700) - San Luis -Argentina
e-mail: rgallard@unsl.edu.ar
Phone: +54 2652 420823
Fax : +54 2652 430224

ABSTRACT

Jobs to be delivered in a production system are usually weighted according to clients requirements and relevance. Attempting to achieve higher customer satisfaction trends in manufacturing are focussed today on production policies, which emphasizes minimum weighted tardiness.

Evolutionary algorithms have been successfully applied to solve scheduling problems. New trends to enhance evolutionary algorithms introduced *multiple-crossovers-on-multiple-parents* (MCMP) a multirecombinative approach allowing multiple crossovers on the selected pool of (more than two) parents. MCMP-SRI is a novel MCMP variant, which considers the inclusion of a stud-breeding individual in a pool of random immigrant parents. Members of this mating pool subsequently undergo multiple crossover operations.

This paper briefly describes the weighted tardiness problem in a single machine environment, and summarizes implementation details and MCMP-SRI performance for a set of problem instances extracted from the OR-Library.

1. INTRODUCTION

The problem $1||\sum w_j T_j$ [1, 4] is an important generalization of the $1||\sum T_j$. Various researchers have worked on this problem and have experimented with many different approaches. These ap-

[1] The Research Group is supported by the Universidad Nacional de La Patagonia Austral.
[2] The LIDIC is supported by the Universidad Nacional de San Luis and the ANPCYT (National Agency to Promote Science and Technology).

proaches range for very sophisticated computer-intensive techniques to fairly crude heuristics designed primarily for implementation purposes. Because branch and bound based method are prohibitively time consuming even with only 20 jobs, it is important to have heuristics providing reasonably good schedules with tolerable computational effort.

Among other heuristics [14], evolutionary algorithms have been successfully applied to solve scheduling problems [15,16]. Current trends in evolutionary algorithms make use of multiparent [5, 6, 7] and multirecombinative approaches [8, 9,10]. The latter we called, *multiple-crossovers-on-multiple-parents* (MCMP). Instead of applying crossover once on a pair of parents this feature applies n_1 crossover operations on a set of n_2 parents. In order to improve the balance between exploration and exploitation in the search process [13], this work make use of a breeding individual (stud) which repeatedly mates individuals that randomly immigrates to a mating pool. Under this approach the random immigrants incorporate exploration (making unnecessary the use of mutation operations) and the multi-mating operation with the stud incorporates exploitation to the search process. Next sections describe the weighted tardiness scheduling problem, the heuristic proposed and discuss the results obtained.

2. THE WEIGHTED TARDINESS SCHEDULING PROBLEM

The single-machine total weighted tardiness problem [1] can be stated as follows: n jobs are to be processed without interruption on a single machine that can handle no more than one job at a time. Job j (j = 1,...,n) becomes available for processing at time zero, requires an uninterrupted positive processing time p_j on the machine, has a positive weight w_j, and a due date d_j by which it should ideally be finished. For a given processing order of the jobs, the earliest completion time C_j and the tardiness $T_j = \max\{C_j - d_j, 0\}$ of job j can readily be computed. The problem is to find a processing order of the jobs with minimum total weighted tardiness $\sum_{j=1}^{n} w_j T_j$. Even with this simple formulation, this model leads to an optimization problem that is NP-Hard [1].

3. MULTIRECOMBINATION OF STUDS AND IMMIGRANTS

The conventional approach to crossover, independently of the method being applied, involves applying the operator only once on the selected parents. Such a procedure is known as the Single Crossover Per Couple approach (SCPC). An alternative approach Multiple Crossover per Couple (MCPC) implies repeated application of crossover to exploit the good features of a pair of parents. Implementation and results are discussed elsewhere [8], [9]. To improve MCPC performance, by using the *multiparent approach* of Eiben [5], [6], [7], the method was extended to MCMP [10] where the multiple crossovers are applied to a set of multiple parents. Results obtained in diverse single and multiobjective optimization problems indicated that the searching space is efficiently exploited by the multiple application of crossovers and efficiently explored by the greater number of samples provided by the multiple parents.

Attempting to achieve a better balance between exploration and exploitation we devised MCMP-SRI [11], [12]. Here, the process for creating offspring is performed as follows (see figure 1). From the old population an individual, designated the stud, is selected by means of proportional selection. The number of n_2 parents in the mating pool is completed with randomly created individuals (random immigrants). The stud mates every other parent, the couples undergo partial mapped crossover (PMX) and 2*n_2 offspring are created. The best of these 2*n_2 offspring is stored in a temporary children pool. The crossover operation is repeated n_1 times, for different cut points each time, until the children pool is completed. Children are not exposed to mutation. Finally, the best offspring created from n_2 parents and n_1 crossover is inserted in the new population.

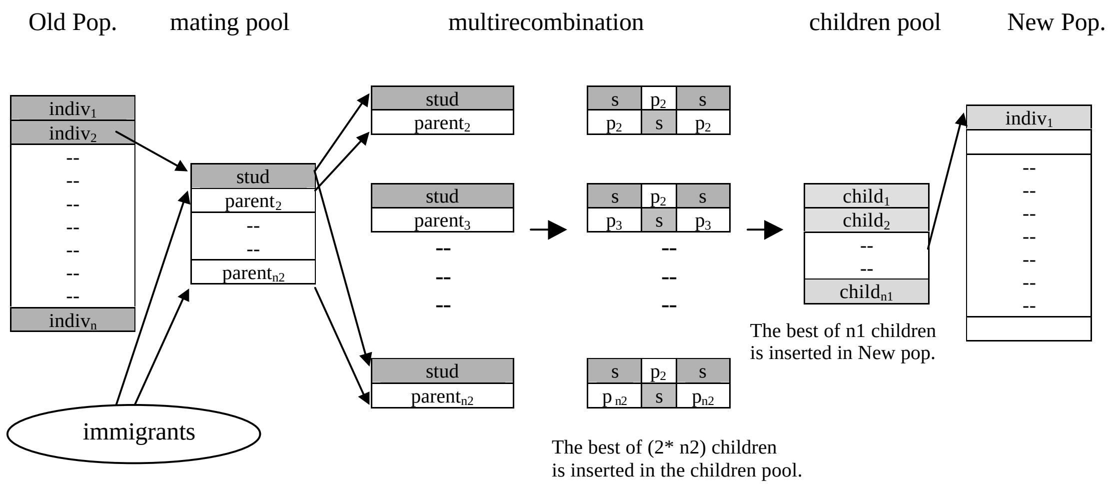

Fig. 1. The stud and random immigrants multirecombination process.

4. EXPERIMENTAL TESTS AND RESULTS

The evolutionary algorithms were tested for selected instances from OR-library benchmarks [2, 3] for the weighted tardiness scheduling problems. We performed a series of 10 runs for each of the 20 instances of 40 and 50 job problems. The maximum number of generations was fixed at 500 and 600 for the 40 and 50 jobs problem size, respectively. Population sizes were fixed at 150 individuals for both problem sizes. Probabilities for crossover were set to 0.65, in all experiments. The number n_1, of crossovers and the number n_2, of parents, were set to 14 and 16 for the 40 and 50 jobs problem size, respectively, in all experiments.

To compare the algorithms, the following relevant performance variables were chosen:

Ebest = ((best value - *opt_val*)/*opt_val*)100
It is the percentile error of the best-found individual when compared with the known, or estimated, optimum value *opt_val*. It gives us a measure on how far the best individual is from that *opt_val*.

Hit Ratio. It is 1 (one) if the algorithm found the reported optimum (or hits the upper bound), and 0 (zero) otherwise.

Gbest. It is the generation where the best individual was found.

In this work we show the results in two tables and four figures for the selected instances. The tables show upper bounds published in the OR-Library, Minimum objective value obtained by MCMP-SRI and the performance variables *Ebest*, *Hit Ratio* and *Gbest*. Figures only show *Ebest* and *Gbest*.

Instance k	Upper Bound	Min WT	Mean Ebest	Hit Ratio	Mean Gbest
WT40-1	913	913	2.16	1	66.30
Wt40-2	6955	6955	0.00	1	220.60
Wt40-3	17465	17465	0.00	1	304.90
Wt40-4	77122	77165	0.50	0	470.40
Wt40-5	77774	77793	0.13	0	477.50
Wt40-6	108	108	0.00	1	71.40
Wt40-7	6575	6575	0.26	1	280.90
Wt40-8	57640	57660	0.73	0	459.30
Wt40-9	64451	64499	0.24	0	478.20
Wt40-10	0	0	0.00	1	51.30
Wt40-11	2099	2099	6.32	1	186.80
Wt40-12	65386	65429	0.58	0	474.70
Wt40-13	90486	90566	0.18	0	485.60
Wt40-14	0	0	0.00	1	4.70
Wt40-15	47683	47811	0.91	0	471.90
Wt40-16	126048	126082	0.12	0	475.90
Wt40-17	0	0	0.00	1	6.60
Wt40-18	0	0	0.00	1	266.30
Wt40-19	46770	46931	0.90	0	461.70
Wt40-20	122266	122458	0.30	0	482.20
Average			**0.67**	**0.50**	**309.86**

Table 1. Mean and general average values for the performance variables under MCMP-SRI, for the 40 jobs instances.

Table 1 summarizes mean and general average values for the performance variables through all selected instances for the 40 jobs problem size. Results show that MCMP-SRI hits the upper bound in half of the runs in the series (Average *Hit Ratio* is 0.50). On average, the percentile error of the best found individual when compared with the best known objective value is 0.67%. The number of generations, *Gbest*, required to find the best individual ranges from 4.70 to 485.60.

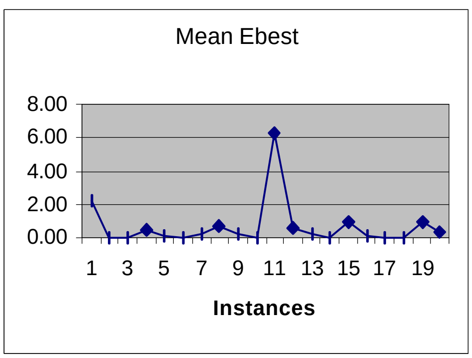

Fig 2. Mean Ebest values under MCMP-SRI, for the 40 jobs instances.

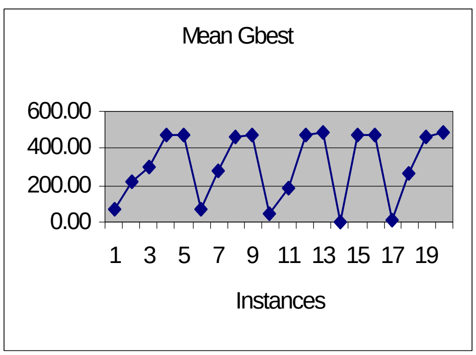

Fig 3. Mean Gbest values under MCMP-SRI, for the 40 jobs instances.

As we can see in figure 2 the mean *Ebest* for most instances is close to zero and only for instance 11 the this value is close to 6%. In figure 3 we can observe that the selected instances require different computational effort (generations) to reach the best found value. The first group of instances requires a number of generations ranging from 0 to 100; for the second group it ranges from 100 to 300 generations and the last group of instances requires more than 300 generations.

Instance k	**Upper Bound**	**Min WT**	**Mean Ebest**	**Hit Ratio**	**Mean Gbest**
wt50-1	2134	2134	0.00	1.00	46.30
wt50-2	26276	26276	0.27	1.00	414.50
wt50-3	51785	51884	0.68	0.00	540.10
wt50-4	89299	90192	1.86	0.00	477.20
wt50-5	214546	215518	1.14	0.00	524.50
wt50-6	2	2	0.00	1.00	224.00
wt50-7	9934	9934	1.06	1.00	417.40
wt50-8	123893	126256	3.02	0.00	476.00
wt50-9	157505	158063	1.64	0.00	494.70
wt50-10	0	0	0.00	1.00	20.00
wt50-11	1258	1258	0.48	1.00	262.80
wt50-12	76878	78215	3.03	0.00	450.80
wt50-13	150580	152213	2.46	0.00	490.20
wt50-14	0	0	0.00	1.00	17.00
wt50-15	89298	91879	5.38	0.00	473.40
wt50-16	177909	178937	1.40	0.00	535.40
wt50-17	0	0	0.00	1.00	15.40
wt50-18	0	0	0.00	1.00	120.70
wt50-19	35727	38354	9.44	0.00	535.00
wt50-20	78315	79595	5.31	0.00	409.00
Mean Avg			**1.86**	**0.45**	**347.22**

Table 2. Mean and general average values for the performance variables under MCMP-SRI, for the 50 jobs instances.

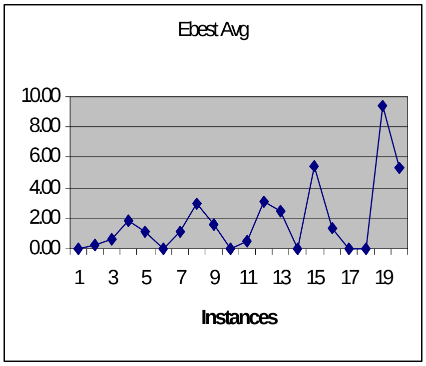

Fig 4. Mean Ebest values under MCMP-SRI, for the 50 jobs instances.

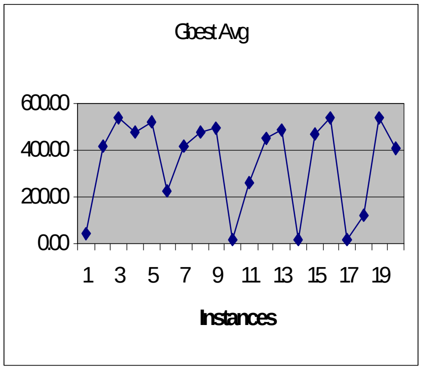

Fig 5. Mean Gbest values under MCMP-SRI, for the 50 jobs instances.

Table 2 summarizes mean and general average values for the performance variables through all selected instances for the 50 jobs problem size. Results show that MCMP-SRI hits the upper bound in approximately half of the runs in the series (Average Mean *Hit Ratio* is 0.45). On average, the percentile error of the best found individual when compared with the best known objective value is 1.86%, while the number of generations, *Gbest*, required to find the best individual ranges from 15.40 to 540.

Figures 4 and 5 show the particular behaviour of *Ebest* and *Gbest* variables for 50 jobs instances. As we can see in figure 4 Mean *Ebest* for most is ranging from 0 to 3 percent and only for instances 15,19 and 20 the *Ebest* is greater than 3%. In figure 5 we can observe that, as it happened in the 40 jobs problem size, the selected instances require different computational effort to reach the best found value. In this case the first group of instances requires a number of generations ranging from 0 to 100; for the second group it ranges from 100 to 400 generations and the last group of instances requires more than 400 generations.

5. CONCLUSIONS

This paper shows the performance of MCMP-SRI, one of the latest variant of the multi-recombinative family when it is applied to the weighted tardiness problem scheduling in a single machine environment. The main objective of this novel recombinative method is to find an equilibrium between exploration and exploitation in the search process. An individual of the old population is selected as the stud and subsequently mated with a set of new randomly generated individuals (immigrants). The presence of the stud ensures the retention of good features of previous solutions while the immigrants, as continuous source of genetic diversity, avoid premature convergence and make it unnecessary to apply mutation. Preliminary results are promising and showed its potentials to find the upper bound in the selected instances of the weighted-tardiness scheduling problem. Future work will include dynamic control and self-adaptation of parameters, and the possible insertion of problem-specific-knowledge in the representation to test the method in the larger benchmarks of the OR library.

6. ACKNOWLEDGEMENTS

We acknowledge the co-operation of the project group for providing new ideas and constructive criticisms. Also to the Universidad Nacional de San Luis, the Universidad Nacional de La Patagonia Austral, and the ANPCYT from which we receive continuous support.

7. REFERENCES

[1] Pinedo, Michael., *Scheduling: Theory, Algorithms and System*. Prentice Hall, first edition, 1995. pp 44-48, 143-145.

[2] Beasley J. E. Common Due Date Scheduling, OR Library, http://mscmga.ms.ic.ac.uk/

[3] Crauwels H.A.J., Potts C.N. and Van Wassenhove L.N.*Local search heuristics for the single machine total weighted tardiness scheduling problem*, Informs Journal on Computing 10, 341-350. 1998.

[4] Chen T. and Gupta M., Survey of scheduling research involving due date determination decision, European Journal of Operational Research, vol 38, pp. 156-166, 1989.

[5] Eiben A.E., Raué P-E., and Ruttkay Zs., *Genetic algorithms with multi-parent recombination*. In Davidor, H.-P. Schwefel, and R. Männer, editors, Proceedings of the 3rd Conference on Parallel Problem Solving from Nature, number 866 in LNCS, pages 78-87. Springer-Verlag, 1994

[6] Eiben A.E., van Kemenade C.H.M., and Kok J.N., *Orgy in the computer: Multi-parent reproduction in genetic algorithms*. In F. Moran, A. Moreno, J.J. Merelo, and P. Chacon, editors, Proceedings of the 3rd European Conference on Artificial Life, number 929 in LNAI, pages 934-945. Springer-Verlag, 1995.

[7] Eiben A.E. and. Bäck Th., An empirical investigation of multi-parent recombination operators in evolution strategies. Evolutionary Computation, 5(3):347-365, 1997.

[8] Esquivel S., Leiva A., Gallard R., - *Multiple Crossover per Couple in Genetic Algorithms*, Proceedings of the Fourth IEEE Conference on Evolutionary Computation (ICEC'97), pp 103-106, ISBN 0-7803-3949-5, Indianapolis, USA, April 1997.

[9] Esquivel S., Leiva A., Gallard R.: *Couple Fitness Based Selection with Multiple Crossover per Couple in Genetic Algorithms*. Proceedings of the International Symposium on Engineering of Intelligent Systems (EIS´98), La Laguna, Tenerife, Spain Vol. 1, pp 235-241, ed. E.Alpaydin. Publishesd by ICSC Academic Press, Canada/Switzerland, February 1998.

[10] Esquivel S., Leiva H.,.Gallard R., *Multiple crossovers between multiple parents to improve search in evolutionary algorithms*, Proceedings of the 1999 Congress on Evolutionary Computation (IEEE). Washington DC, pp 1589-1594.

[11] Pandolfi D., Vilanova G., De San Pedro M., Villagra A. Multirecombining studs and immigrants in evolutionary algorithm to face earliness-tardiness scheduling problems. Proceedings of the International Conference in Soft Computing. University of Paisley, Scotland, U.K., June2001.

[12] Pandolfi D.; Vilanova G.; De San Pedro M.; Villagra A. Gallard R.: *Solving the Single-Machine Common Due Date Problem Via Stud and Inmigrants in Evolutionary Computation* vol III pp 409-413 World Multiconference on Systemics, Cybernetics and Informatics Orlando 2001

[13] Michalewicz, M., *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, third revised edition, 1996.

[14] Morton T., Pentico D., *Heuristic scheduling systems*, Wiley series in Engineering and technology management. John Wiley and Sons, INC, 1993.

[15] Reeves C., *A genetic algorithm for flow shop sequencing*, Computers and Operations Research, vol 22, pp5-13, 1995.

[16] Tsujimura Y., Gen M., Kubota E.: *Flow shop scheduling with fuzzy processing time using genetic algorithms*. The 11th Fuzzy Systems Symposium pp 248-252. Okinawa. 1995.